## REMARKS

Claims 1-20 and 22-24 are currently pending in the subject application and are presently under consideration. 1, 15, 17, 19 and 22-24 have been amended as shown on pp. 2 and 4-7 of the Reply. Favorable reconsideration of the subject patent application is respectfully requested in view of the comments and amendments herein.

## I. Rejection of Claims 1-6, 8-13, 15, 16, and 22-24 Under 35 U.S.C. §103(a)

Claims 1-6, 8-13, 15, 16, and 22-24 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Necula, *et al.* (US 6,128,774) in view of Stata, *et al.* (US 6,353,925) and in view of Hallem, *et al.* ("A System and Language for Building System-Specific, Static Analyses"). Withdrawal of this rejection is requested for at least the following reasons. Necula, *et al.* either alone or in combination with Stata, *et al.* and/or Hallem *et al.*, does not teach or suggest every aspect of the subject claims.

> ***[T]he prior art reference (or references when combined) must teach or suggest all the claim limitations.*** *See* MPEP § 706.02(j). *See also KSR Int'l Co. v. Teleflex, Inc.*, 550 U. S. ____, 04-1350, slip op. at 14 (2007). The teaching or suggestion to make the claimed combination and the reasonable expectation of success must be found in the prior art and not based on applicant's disclosure. *See In re Vaeck,* 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991) (emphasis added).

Applicants' claimed subject matter relates to a system and method that can employ a framework for declaration of preconditions and/or postconditions. In particular, the precondition and/or postcondition specifications can include rules for using an interface, system resource management, order of method calls and/or formatting of string parameters. Thus, the system can mitigate run-time exceptions associated with conventional systems, such as exceptions associated with improper management of system resources, order of method calls and/or formatting of string parameters (*e.g.*, SQL queries). In particular, independent claims 1, 15, 22-24, as amended, recite similar aspects, namely, ***the plug-in condition includes one or more rules for using an interface, system resource management, order of method calls and formatting of string parameters.*** The cited references, alone or in combination, fail to teach or suggest this novel aspect.

Necula, *et al.* relates to a system that verifies safe execution of software. In particular, the system, verifies that untrusted software supplied by a code producer is safe to execute by a code consumer. Specifically, the code consumer can declare a precondition, stored in the configuration data, and guarantee that the precondition holds when the untrusted code is invoked. (*See* column 7, lines 19-22). More specifically, the precondition is essentially a description of the calling convention the consumer will use when invoking the untrusted code. For example, if the untrusted code needs to access Ethernet network packets, the code consumer might declare that the first argument passed to the code is an array of length at least 64 bytes. With this assumption, array accesses to most packet header fields can be proved safe without the need for run-time array-bounds checking. Further, the safety policy can also declare postconditions for the untrusted code that are constraints on the final execution state of the untrusted code. (*See* column 5, lines 50-65). However, Necula, *et al.* is silent with respect to enabling a programmer to declare plug-in conditions that can facilitate detection of errors in utilizing an interface, system resource management, order of method calls and formatting of string parameters.

Stata, *et al.* relates to a system and method wherein tool-specific annotations are recognized by the lexer for the programming-language, but the lexing and parsing of the tool-specific annotations are handled by a separate, tool-specific annotation processor. In particular, a selected annotation processor converts annotations compatible with the user-selected tool into annotation tokens and returns the annotation tokens to the lexer. The lexer generates tokens based upon the programming-language statements in the source program, and passes both the tokens and annotation tokens to a parser. Further, the parser, in turn, assembles the tokens and annotation tokens into an abstract syntax tree, which is then passed to the user-selected tool for further processing. However, Stata, *et al.* fails to disclose a plug-in condition that includes one or more rules for using an interface, system resource management, order of method calls and formatting of string parameters and thus fails to make up for the deficiencies of Necula, *et al.* discussed *supra.*

Further, Hallem, *et al.* simply relates to a system and language for building system-specific static analysis. The language allows users to specify a broad class of analyses in terms that resemble the intuitive description of the rules they check. However, the system disclosed by Hallem, *et al.* fails to teach or suggest a plug-in condition that includes one or more rules for

using an interface, system resource management, order of method calls and formatting of string parameters as disclosed by the subject claims.

Applicants' claimed subject matter discloses a system and method that employing pre- and/or post- conditions specified at a source code level and persisted (*e.g.,* in associated object code and/or a specification repository) facilitating static checking of the object code is provided. A specifier can give a method a plug-in pre- and postcondition, which is arbitrary code that examines an object's current state and a static approximation of the method's actuals, decides whether the call is legal and returns the object's state after the call. In contrast to the cited references, the subject system provides a declarative framework that allows rules for using an interface to be recorded as declarative specifications and provides a range of annotations that allow a developer to specify interface rules with varying precision. At the simplest end of the range, a specifier can mark those methods that allocate and release resources. A specifier can also limit the order in which an object's methods may be called to the transitions of a finite state machine. At the more complex end of the range and in accordance with an aspect of the present invention, a specifier can give a method a plug-in pre-and postcondition -- code (*e.g.,* arbitrary) that examines an object's current static state and a static approximation of the method's actuals, decides whether the call is legal and returns the object's state after the call. (*See* page 8, lines 12-21.) Conventional compilers do not check protocols for interfaces, even in a safe programming language, such as C# or Java, disobeying the rules for using an interface can cause exceptions at run time. Such rules govern how system resources are managed, the order of method calls, and the formatting of string parameters, such as SQL queries and XML element tags. Applicants' subject specification discloses a system and method that enables the specifier to verify these rules. The cited references, alone or in combination, fail to teach or suggest these novel features.

In view of at least the foregoing, it is readily apparent that Necula, *et al.* alone or in combination with Stata, *et al.* and/or Hallem, *et al.* fails to teach or suggest all features of applicants' invention as recited in independent claims 1, 15 and 22-24 (and associated dependent claims), and thus fails to make obvious the subject claims. Accordingly, withdrawal of this rejection is respectfully requested.

## II. Rejection of Claim 7 Under 35 U.S.C. §103(a)

Claim 7 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Necula, *et al.* (US 6,128,774) in view of Stata, *et al.* (US 6,353,925) and Hallem, *et al.* ("A System and Language for Building System-Specific, Static Analyses"), as applied to claim 1, and further in view of Meijer, *et al.* ("Technical Overview of the Common Language Runtime"). Withdrawal of this rejection is requested for at least the following reasons. Necula, *et al.*, Stata, *et al.* and Hallem, *et al.*, either alone or in combination with Meijer, *et al.*, do not teach or suggest every aspect of the subject claims.

Claim 7 depends from independent claim 1. As discussed *supra*, Necula, *et al.*, Stata, *et al.* and/or Hallem, *et al.* do not teach all aspects of independent claim 1. In particular, Necula, *et al.*, Stata, *et al.* and/or Hallem, *et al.* are silent with regard to specification of rules that govern how system resources are managed, the order of method calls, and the formatting of string parameters. Meijer, *et al.* discloses a Common Language Runtime (CLR) that is expressed in the Common Intermediate Language (CIL), can be compiled from a language such as C, Pascal, C# etc. However, Meijer, *et al.* fails to cure the aforementioned deficiencies presented by Necula *et al.*, Stata, *et al.* and/or Hallem, *et al.* with respect to the independent claim 1. Accordingly, it is respectfully requested that the rejection of claim 7 be withdrawn.

## III. Rejection of Claim 14 Under 35 U.S.C. §103(a)

Claim 14 stands rejected under 35 U.S.C. §103(a) as being unpatentable over Necula, *et al.* (US 6,128,774) in view of Stata *et al.* (US 6,353,925) and Hallem, *et al.* ("A System and Language for Building System-Specific, Static Analyses"), as applied to claim 1, and further in view of Goldberg, *et al.* (US 6,571,232). Withdrawal of this rejection is requested for at least the following reasons. Claim 14 depends on independent claim 1. As per the above discussion, Necula, *et al.*, Stata, *et al.* and/or Hallem, *et al.* do not teach or suggest each and every feature of claim 1. Goldberg, *et al.* merely relates to a query object generator tool that generates interface definitions and code that implement a query object also generates a database schema access query object that retrieves the schema of an underlying database but fails to remedy the aforementioned deficiencies presented by Necula, *et al.*, Stata, *et al.* and/or Hallem, *et al.* with respect to independent claim 1. Thus, withdrawal of this rejection is respectfully requested.

## IV. Rejection of Claims 17 and 18 Under 35 U.S.C. §103(a)

Claims 17 and 18 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Tichelaar, *et al.* ("A Metal-model for Language-Independent Refactoring") in view of Stata, *et al.* (US 6,353,925) and in view of Hallem, *et al.* ("A System and Language for Building System-Specific, Static Analyses"). Withdrawal of this rejection is requested for at least the following reasons. Tichelaar, *et al.,* alone or in combination with Stata, *et al.* and/or Hallem, *et al.* does not disclose all features recited in the subject claims. In particular, independent claim 17, as amended, recites ***the at least one of a plug-in precondition or a plug-in postcondition includes at least one rule for using an interface, system resource management, order of method calls and formatting of string parameters***. The cited references are silent with respect to this novel aspect.

As discussed above, Stata, *et al.* and/or Hallem, *et al.* fail to teach or suggest rules that govern how system resources are managed, the order of method calls, and the formatting of string parameters. Tichelaar, *et al.* relates to a language independent refactory engine that changes a system to improve its internal structure without altering its external behavior. The engine provides standard method for programmers and tools to perform refectories no matter which language they work in. However, Tichelaar, *et al.* fails to cure the deficiencies of Stata, *et al.* and Hallem, *et al.* with respect to independent claim 17. Thus, it is readily apparent that Tichelaar, *et al.*, Stata, *et al.* and Hallem, *et al.,* alone or in combination, do not anticipate or suggest the subject invention as recited in claim 17 (and claim 18 that depends therefrom). Accordingly, withdrawal of this rejection is respectfully requested.

## V. Rejection of Claims 19 and 20 Under 35 U.S.C. §103(a)

Claims 19 and 20 stand rejected under 35 U.S.C. §103(a) as being unpatentable over Necula, et al. (US 6,128,774) in view of Stata, *et al.* (US 6,353,925). Withdrawal of this rejection is requested for at least the following reasons. Necula, *et al.* either alone or in combination with Stata, *et al.,* does not teach or suggest every aspect of the subject claims. In particular, independent claim 19 has been amended to recite ***the precondition plug-in includes one or more rules for using an interface, system resource management, order of method calls and formatting of string parameters.*** As discussed *supra*, with respect to independent claims 1, 15 and 22-24, that recite similar aspects, Necula, *et al.* and/or Stata, *et al.*, are silent with respect

to rules that govern how system resources are managed, the order of method calls, and the formatting of string parameters. Additionally, independent claim 19 recites ***removing the precondition plug-in from the executable code to reduce the overall physical storage requirements associated with the executable code.*** Specifically, a method is disclosed in the subject specification wherein source code for a software component is created (*e.g.,* by a developer) employing the framework for declaration of a plug-in condition (*e.g.,* preconditions and/or postcondition) specifications at the source code level. The source code is then compiled into executable code (*e.g.,* object code) with the plug-in condition (*e.g.,* precondition and/or postcondition) specifications being embedded within the executable (*e.g.,* object code). The executable code (*e.g.,* object code) with embedded specifications is received by an input component and is provided to the checker. The checker employs the specification to facilitate static checking of the object file provides information (*e.g.,* to the developer) if a fault condition (*e.g.,* errors) is determined to exist. More specifically, once the checker has employed the embedded specification, the embedded specification can be removed from the executable code, thus, reducing the overall physical storage requirements associated with the executable code. (*See* page 8 line 22- page 9 line 4.) Necula, *et al.,* alone or in combination with Stata, *et al.*, fails to teach these novel aspects. Therefore, it is respectfully requested that this rejection be withdrawn.

## CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments and amendments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063 [MSFTP482US].

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

AMIN, TUROCY & CALVIN, LLP

/Himanshu S. Amin/
Himanshu S. Amin
Reg. No. 40,894

AMIN, TUROCY & CALVIN, LLP
24TH Floor, National City Center
1900 E. 9TH Street
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731